John T. Anderson

Engineering Note

Date: September 27th, 2002

Rev Date: October 8th, 2002

Project: Central Fiber Tracker
Doc. No: A1020927

Subject: Diagnostic and functional additions to the VSVX in response to trigger formation time increases

Introduction

Current trigger timing requirements in the Central Fiber Tracker (CFT) have necessitated changes to the Sequencer code such that the SVX clock during the Acquire mode is now no longer a simple clock-every-crossing, but instead a clock which has alternating 264 nsec and 132 nsec periods. The trigger formation time has steadily increased until the system is now in the 33rd pipeline out of a maximum of 31, a difficult condition. So long as the trigger was held in the "32nd" pipeline, one crossing of delay implemented in the SIFT was sufficient to solve the problem. Now, however, the delay increase forces a redefintion of the SVX clock. This complicates the calculation of the correct SVX Pipeline number. In addition, the VSVX Pipeline Delay number has overflowed the maximal value originally designed.

Total Pipeline Delay Solution

To solve the two issues associated with the longer triggering time, the VSVX CPLD has been modified. The overflow of the pipeline number is trivially solved by a minor change to the event delay state machine. Instead of delaying 'N' ticks of the 61 MHz clock as originally programmed, the machine now delays '2N-1' clocks. In both forms, 'N' is a number programmed externally via 1553. This is easily accomplished by changing the delay loop to enclose two states of the machine rather than the one state formerly used. The following snippet of code from the *EV_DLY.ABL* source file highlights the changes. The new version of code is stored in the *132_tickcnt_20020923* project folder for the VSVX. Lines specific to the changes have been highlighted.

```
"
"       beginning of Acquire mode sequence.  When the SVX is put into Acquire
"       mode the machine simply counts the number of clocks since Acquire mode
"       starts.  At the appropriate value it enables the read clock to keep the depth
"       of the event delay fifo pipeline constant.  The pipeline delay count is a count
"       of 61 MHZ clocks.  There are eight counts per crossing, which means that the
"       counter matches the depth of the real SVX at 32 crossings.
"
"       The read CLOCK is turned on early to allow it to run cleanly (no startup issues).
"       The timing control is handled with the read ENABLE.
"
"       Modified 9/11/02 to change acquire loop to delay DOUBLE the pipe count
"
STATE ACQ1:
        EV_FIFO_RE := 1;        // disable reads
        EV_FIFO_RCLK = 0;       // start with read clock off to be safe
        COLLECTOR_WCLK := 0;    // don't put anything into the collector FIFO
        ACQ_CNTR := ACQ_CNTR;   // do-nothing
```

```
        DIG_CNTR := DIG_CNTR;  // counter not used
        OE_SR_RST = 1;         // hold oe pattern in reset state
        ADV_OE = 0;            // do not clock oe pattern
        GOTO ACQ1A;            // count off the initial delay
"
"       In state ACQ1A, enable the clock.
"
STATE ACQ1A:
        EV_FIFO_RE := 1;       // disable reads
        EV_FIFO_RCLK = XCLK_69;       // turn on the read clock
        COLLECTOR_WCLK := 0;   // don't put anything into the collector FIFO
        ACQ_CNTR := PIPE_DELAY;       // load the count
        DIG_CNTR := DIG_CNTR;  // counter not used
        OE_SR_RST = 1;         // hold oe pattern in reset state
        ADV_OE = 0;            // do not clock oe pattern
        GOTO ACQ2;             // and start countin'

"
"       In state ACQ2, load the delay counter with the programmed value.
"       if we need to offset the delay by one, put the exit condition test
"       in ACQ2A and make ACQ2 just a fallthrough goto.
"
STATE ACQ2:
        EV_FIFO_RE := 1;       // disable reads
        EV_FIFO_RCLK = XCLK_69;       // let read clock run as we delay
        COLLECTOR_WCLK := 0;   // don't put anything into the collector FIFO
        ACQ_CNTR := ACQ_CNTR;  // hold the value
        DIG_CNTR := DIG_CNTR;  // counter not used
        OE_SR_RST = 1;         // hold oe pattern in reset state
        ADV_OE = 0;            // do not clock oe pattern
        IF (ACQ_CNTR == 0) THEN ACQ3;// advance into rest of acquire loop when counter hits zero
        IF (ACQ_CNTR != 0) THEN ACQ2A;  //otherwise continue (two 61 mhz clocks per pipe count)

"
"       In state ACQ2A, simply count down the counter until it hits zero.  At that point
"       turn on the read enable and enter state ACQ3.
"
STATE ACQ2A:
        EV_FIFO_RE := 1;       // disable reads
        EV_FIFO_RCLK = XCLK_69;       // let read clock run as we delay
        COLLECTOR_WCLK := 0;   // don't put anything into the collector FIFO
        ACQ_CNTR := ACQ_CNTR -1;      // count down
        DIG_CNTR := DIG_CNTR;  // counter not used
        OE_SR_RST = 1;         // hold oe pattern in reset state
        ADV_OE = 0;            // do not clock oe pattern
```

```
        GOTO ACQ2;              // continue in loop
"
"       State ACQ3 turns on the read enable, because we've delayed long enough, and
"       keeps the FIFO at the current delay depth by reading as fast as the fifo is being
"       filled.  The read enable is driven by the VSVX_PARST signal, which comes from the
"       CLOCKGEN pld.  This signal is turned on when Acquire mode starts and persists until
"       the Sequencer gives the prompt L1_ACCEPT.  This will precede the exit from Acquire
"       mode, so the machine has to stay in state ACQ3 until Acquire mode ends; however, the
"       FIFOs have to stop flushing promptly upon the receipt of L1_ACCEPT.
"
STATE ACQ3:
        EV_FIFO_RE := !VSVX_PARST;    // enable reads for as long as VSVX_PARST is asserted
        EV_FIFO_RCLK = XCLK_69;       // continue to let read clock run
        COLLECTOR_WCLK := 0;  // don't put anything into the collector FIFO
        ACQ_CNTR := ACQ_CNTR; // counter not used
        DIG_CNTR := DIG_CNTR; // counter not used
        OE_SR_RST = 1;        // hold oe pattern in reset state
        ADV_OE = 0;           // do not clock oe pattern
        IF (ACQ_MODE) THEN ACQ3;  // stay waiting here until Acquire mode is over
        ELSE IDLE;
```

Additional Diagnostic Features

The complexity of the new SVX timing structure and the significant risk of glitches or timing problems associated with Sequencer changes (an unfortunate feature of Altera chips when they get rather full) requests additional timing diagnosis from the AFE. Fortunately the VSVX has a relatively free byte of information in the Status byte. Historically the Status byte from the VSVX has simply been an echo of the control value written to the device. The Status byte from SVX chips has always been set to zero, so event unpackers typically ignore this value entirely. Previous elimination of Monitor FIFO features from disuse has rendered all but two of the bits in the VSVX control value free; thus, six bits are available to control new diagnostic features. The new version of the VSVX redefines the control register to have the following bit definitions:

Bit	Name	Function
7	Spare	Bit not used in this design.
6	SKIP_OTHER_CHIPS	If set, only VSVX reads out, SVX chips are skipped (as in previous releases)
5	Spare	Bit not used in this design.
4	TCNT_CSEL1	Time Counter Clock Select bit 1
3	TCNT_CSEL0	Time Counter Clock Select bit 0
2	ENABLE_SIFT	If set, VSVX reads out discriminator bits; if not, only Chip Id/Status is issued (as in previous releases)
1	TCNT_RSEL1	Time Counter Reset Select bit 1
0	TCNT_RSEL0	Time Counter Reset Select bit 0

Status Word as Time Counter

The essential change to the VSVX is to replace the Status word by a Time Counter that gives diagnostic information regarding the Sequencer operation during the previous Acquire/Digitize sequence when the VSVX is read out. The counter is implemented as an eight-bit binary up counter with an Enable, a Reset, and a Clock. The Enable definition is fixed - the counter counts continuously during the Acquire sequence until the L1_Accept is received from the Sequencer. At that time the counter is disabled, re-enabling at the end of the VSVX's portion of the Readout cycle. The user may select various clock speeds and Reset conditions to measure different times of interest.

Reset Conditions

Two bits (1 and 0) of the control byte select one of four conditions under which the Time Counter is reset. The reset may occur many times before the counter is held, so that only the last timing measurement is preserved for readout. The following four conditions are defined:

Bit pattern	Reset Function
00	First Crossing during Acquire Mode
01	Falling Edge of Sync Gap
10	Rising Edge of Sync Gap
11	GREEN_WIRE(3), carrying the Beginning-of-Turn mark from the Clockgen state machine.

These reduce to the following measurements:

- 00 measures the time from the last occurrence of First Crossing in the Acquire Mode to when the L1_Accept is received. This should be a constant number when testing with the SASEQ, and presumably also when test triggers are in use with the Sequencer. Of course, for real accelerator data, one would expect a flat distribution of numbers - with any significant peaks indicating a potential bias in the trigger.
- 01 measures a similar time, but uses the falling edge of the Sync Gap.
- 10 gives the required double-check if 00 and 01 give variant results, allowing direct measurement of the time at which the Sync Gap signal rises. By inference one may then calculate the width of the Sync Gap (presuming it is consistent across multiple readouts!).
- 11 measures the time from the Clockgen's Beginning-of-Turn marker, which should be the same as (First Crossing + (7 * 132 nsec)). This is because the Clockgen marks the *actual* beginning-of-turn (defined as tick #0), and the First Crossing mark from the Sequencer is "the first crossing after the Sync Gap ends". Using this condition allows one to check that the Clockgen is working, and also by inference, if the First Crossing mark from the Sequencer is drifting - since the Clockgen ignores the First Crossing signal sent by the Sequencer and synchronizes to the Sync Gap instead.

In general, the reset conditions all have to do with measuring how well the Sequencer stays in sync.

Clock Control

Complementary to the reset selection, bits 4..3 of the control word allow selection of one of four *clocks* to the Time Counter, allowing measurement of the selected reset-to-stop time in four different ways. The selections provided are:

Bit pattern	Clock Function
00	Crossing Clocks
01	SVX Clocks
10	61 MHz Clocks
11	Beam_396 Clocks

The different clock selections allow direct measurement of how the VSVX and the SVX should 'see' the same section of time. Selecting Crossing Clocks gives a direct measurement - in 132 nsec ticks - of the selected time range. Selecting SVX clocks should give a direct measurement of the *actual* SVX pipeline number that *should* be programmed into the SVX. This provides direct confirmation of the 264/132 nsec mode recently put into the Sequencer. Using 61 MHz clocks gives appropriately finer resolution, at the cost of reducing the available measurement range to 255 counts (about 4.2 usec, or just short of 32 crossings). This isn't long enough to directly measure the normal triggering times, but can be used as a vernier on top of the measurement using the direct crossing clock – just don't forget the possibility of rollover. For good measure, the beam marker is thrown in, just as a cross-check against the SVX clock.

Note that this system is not immune to bogus clocks delivered to the AFE. For example, if the Crossing Clock (supposed to be a fixed and unchanging one-pulse-every-132-nsec) has a glitch or misses a tick, this won't cause the system to capture a count. Similarly, this setup doesn't capture changes in the width or position of the Sync Gap. The system only captures the *last* timing interval prior to the L1_Accept. Erroneous clocks occuring randomly elsewhere cannot be seen.